CLAIMS

1. A method of decoding a bitstream encoded according to a Huffman coding tree of height H comprising:

extracting a first codeword of H bits from the bitstream;

modifying the codeword by shifting it by a first shift value;

using this modified codeword to identify using at least a first data structure either a symbol or a second data structure having an associated second offset value and an associated second shift value; and

if a second data structure is identified using the first data structure:

> modifying the codeword by subtracting the second offset value and shifting the result by the second shift value; and
>
> using this modified codeword to identify using the second data structure either a symbol or a third data structure having an associated third offset value and an associated third shift value.

2. A method as claimed in claim 1, further comprising accessing a look-up table to obtain the first shift vale and accessing the look-up table to obtain the second offset value and the second shift value.

3. A method as claimed in claim 1 or 2, wherein the first data structure represents a first level of the Huffman coding tree and the second data .structure represents a second, lower level of the Huffman coding tree.

4. A method as claimed in any one of claim 1 to 3, further comprising receiving at least the value of height H, the first shift value, the second offset value, the second shift value, the first data structure and the second data structure.

5. A method as claimed in any one of claims 1 to 4, wherein the step of modifying the codeword by shifting it by a first shift value comprises firstly subtracting a first off-set value , if any, from the codeword and then shifting the result by the first shift value.

6. A computer program for performing the method of any one of claims 1 to 5.

7. A storage medium or transmission medium embodying the computer program of claim 6.

8. A method of decoding a bitstream encoded according to a Huffman coding tree of height H comprising:
extracting a codeword of H bits from the bitstream;
shifting the codeword by a predetermined shift value; and
using the modified codeword to identify a symbol using at least a first data structure.

9. A method as claimed in claim 7, further comprising accessing a look-up table to obtain the predetermined shift vale.

10. A method as claimed in claim 8 or 9, wherein the first data structure represents a first level of the Huffman coding tree.

11. A method as claimed in any one of claim 8 to 10, further comprising receiving at least the value of height H, the predetermined shift value, and the first data structure.

12. A method as claimed in any one of claims 8 to 11, wherein the step of shifting the codeword by a predetermined shift value comprises firstly subtracting a first off-set value , if any, from the codeword and then shifting the result by the predetermined shift value.

13. A computer program for performing the method of any one of claims 8 to 5.12

14. A storage medium or transmission medium embodying the computer program of claim 13.

15. A decoder for decoding a bitstream encoded according to a Huffman coding tree of height H comprising:
a memory for storing a plurality of data structures representing the Huffman coding tree of height H including at least a first data structure having an associated first offset value and an associated first shift value and a second data structure having an associated second offset value and an associated second shift value; and
a processor operable to
subtract a current offset value from a codeword of H bits taken from the bitstream;

shift the result by the associated shift value; and
address the associated data structure using the result.

16. A decoder as claimed in claim 15, wherein the first data structure represents a first level of the Huffman coding tree and the second data structure represents a second, lower level of the Huffman coding tree.

17. A decoder as claimed in claim 16, wherein the first shift value corresponds to the first level.

18. A decoder as claimed in claim 16 or 17, wherein the second shift value corresponds to the second level.

19. A decoder as claimed in claim 16, 17 or 18 wherein the second offset value identifies a position of a first sub-tree within the Huffman tree.

20. A decoder as claimed in any one of claims 17 to 19, wherein the processor is operable having obtained a value from addressing the associated data structure, to perform a comparison using that value and in dependence upon the comparison either use the value to identify a symbol or a new current offset value.

21. A decoder as claimed in claim 20, wherein the comparison uses the MSB of the value.

22. A decoder as claimed in claim 20 or 21, wherein the current offset value is initially set to the first offset value.

23. A method of decoding a bitstream encoded according to a Huffman coding tree of height H comprising:
storing a first data structure comprising a value for each possible node at a first level of the tree;
storing a second data structure comprising a value for each possible node within a first sub-tree at a second, lower level of the tree;
extracting a first codeword of H bits from the bitstream;

converting the value of the first codeword into a first node position within the tree at the first level of the tree; and

accessing the first data structure to obtain the value corresponding to the first node position, wherein that value refers to the second data structure;

converting the value of the first codeword into a second node position within the first sub-tree at the second level of the tree; and

accessing the second data structure to obtain the value corresponding to the second node position.

24. A computer program for performing the method of claim 23.

25. A storage medium or transmission medium embodying the computer program of claim 24.

26. A method of decoding a codeword from a bit stream comprising:

receiving a representation of a Huffman tree as a plurality of ordered data structures comprising: a first data structure associated with an identified first level L1 of the tree and comprising a plurality of data entries, each entry corresponding to a node of a full tree at the identified first level and at least a second data structure associated with an identified second level L2 of the tree and with an identified first sub-tree and comprising a plurality of data entries, each entry corresponding to a node of the first sub tree, when full, at the second identified level;

obtaining a value for a first level L1 in a Huffman tree

identifying the node in the first level L1 of the tree, when full, corresponding to the first L1 bits of the codeword;

obtaining from the first data structure a data entry for the identified node, that identifies a further data structure if the identified node is an interior node and otherwise identifies a symbol; and

if the identified node is an interior node:

> obtaining a value for a second level L2 in a Huffman tree, being a higher level than the first level L1;
>
> obtaining a value identifying a first sub-tree;
>
> identifying the node in the second level L2 of the first sub-tree, when full, corresponding to the first L2 bits of the received bit stream;

obtaining from a further data structure a data entry for the identified node, that identifies a further data structure if the identified node is an interior node and otherwise identifies a symbol.

27. A computer program for performing the method of claim 26.

28. A storage medium or transmission medium embodying the computer program of claim 27.

29. Data representing a Huffman coding tree comprising leaf nodes and interior nodes arranged in H levels, wherein each leaf node depends from a single interior node on the next lowest level and represents a symbol and each interior node depends from a single interior node on the next lowest level, the data comprising:
a first data structure identifying, for each of the nodes within a first specified level of the tree, a symbol for each leaf node and a further data structure for each interior node, including a second data structure for a first interior node;
at least a second data structure, identified by the first data structure, identifying for each of the nodes within a sub-tree, depending from the first interior node, and at a second specified level of the tree, a symbol for each leaf node and a further data structure for an interior node, if any; and
data specifying at least the first level, the second level and the first interior node.

30. Data as claimed in claim 29, wherein the first data structure identifies a symbol for each empty node, if any.

31. Data as claimed in claim 29 or 30, wherein the second data structure identifies a symbol for each empty node of the sub-tree at a second level of the tree,

32. Data as claimed in claim 29, 30 or 31 wherein the first level is the lowest level within the tree with at least two leaf nodes.

33. Data as claimed in any one of claims 29 to 32, wherein the second level is the lowest level within the sub-tree with at least two leaf nodes.

34. Data as claimed in any one of claim 29 to 33, wherein the first interior node, when at level L (L=0, 1,2..) and having a value V, is specifying by a value dependent upon V*2^(H-L).

35. Data as claimed in any one of claims 29 to 34, further comprising data specifying H.

36. A storage medium or transmission medium embodying the data as claimed in any one of claims 29 to 25.

37. A method of representing a Huffman binary tree comprising:
producing a first data structure associated with an identified first level L1 of the tree and comprising a plurality of data entries, each entry corresponding to a node of a full tree at the identified first level and identifying a further data structure if that node is an interior node and otherwise identifying a symbol; and
producing at least a further data structure associated with an identified second level L2 of the tree and with an identified first sub-tree and comprising a plurality of data entries, each entry corresponding to a node of the first sub tree, when full, at the second identified level L2 and identifying a further data structure if that node is an interior node and otherwise identifying a symbol.

38. A method as claimed in claim 37, running an algorithm to determine the number of data structures and their associated levels within the Huffman tree.

39. A method as claimed in claim 37 or 38 further comprising identifying a sub-tree having a root node at level L (L=0, 1,2..) and value V using a value dependent upon V*2^(H-L).

40. A method of encoding and/or a method of decoding substantially as hereinbefore described with reference to and/or as shown in the accompanying drawings.

41. An encoder or decoder substantially as hereinbefore described with reference to and/or as shown in the accompanying drawings.